

REMARKS

Claims 1-21 are pending, Claims 1, 8 and 15 have been amended, and new Claims 32-37 are hereby requested to be entered. Claims 22-31 were withdrawn by the Examiner in the Office Action dated June 3, 2002. No new matter has been added. Support for the amended claims and new claims may be found in the application at least at pages 16, 23, 137-138, 177-178, and 183-186.

Rejection under 35 U.S.C. § 102


Claims 1-21 were rejected under 35 U.S.C. § 102(b) as being anticipated by Lau (U.S. Patent No. 5,987,247). Applicants submit that the present amendments overcome that rejection.

The presently claimed invention has been amended to be directed to a computer program for handling tasks to be carried out by an employee that "allows a user to input rules which dictate which of the tasks should be selected based on predetermined events." Thus, the claimed invention is directed to a workflow management computer program for directing the activities of employees assigned to carry out tasks that achieve a goal upon completion, where employee-input rules dictate which tasks are selected based on predetermined events. One embodiment of this computer program is described as a "server based framework utilizing component based architecture." (application, p. 16). This framework is the

computer program for handling tasks as described in the application to which the claims are now directed.

In contrast to Lau, Claims 1, 8 and 15 are directed not to system development software to generate a program as in Lau, but to the business application program, which is generated by programmers and/or system developers. In this context, the users are not the programmers/system developers, but are the employees who perform the business functions and carry out tasks to complete a goal supported by the computer program. The goal may be an insurance related goal, as claimed in Claims 6, 13 and 20. Some of those employees may be assigned the authority of a Task Librarian who may use the computer program also to input new rules and new tasks to modify the business logic design of the system without having to modify the computer program code. For example, the present application at page 184, lines 5-26, describes a preferred embodiment of how the program creates a user interface to allow the users to input tasks and rules into the Task Library database 1500 to achieve a flexible system in which business processes may be quickly changed without recoding the program.

In contrast to the claimed invention, Lau fails to disclose or suggest a computer program for handling tasks that also is adapted to "allow a user to input rules which dictate which of the tasks should be selected based on predetermined events" as required by Claims 1, 8 and 15. Applicants also submit that Lau does not teach a "client component" or "a computer program for handling tasks to be



carried out by employees" which allows a user to input the rules for determining which tasks are selected based on predetermined events. Applicants submit Lau does not teach that the software code created to carry out those rules is itself adapted to allow a user to input those rules by using the software code itself or a client component of the software. In other words, Lau teaches that a programmer or developer would first need to obtain the rules from the business personnel and then use the software development system of FIG. 3 (a first computer program) to generate software code (a second computer program) to carry out those rules. (col. 8, lines 29-31 and 46-67).

In the Office Action, the Examiner asserted that "the computer program developed by the system of Lau can handle tasks, in particular the task of handling insurance claims." and that "FIG. 5, pane 2 allows a user to define rules." (page 3). However, the present claims require that the computer program both handle tasks and allow the user to define the rules. In contrast to the claims, it is not the program for handling tasks generated by the system of Lau, but the system of Lau itself that allow users to define rules in the business logic design (301).

Moreover, applicants dispute the Examiner's characterization of Lau with regard to the developer interface of FIGs. 4 and 5. In contrast to the Examiner's assertions in the Office Action dated May 9, 2001 (Paper No. 4), Fig. 5, pane 3 (503) does not define tasks to be carried out by an employee, but merely lists the methods for the "getter" and "setter" of attributes of an interface selected by a

developer. These are not examples of the claimed tasks. Pane 2 (502) does not define the rules that control the tasks, but merely provides a graphical view of relationships and inheritance between data objects and does not provide for entry of rules as required by the claims. Likewise, pane 1 (501) does not disclose a predetermined event which invokes a rule, but merely is a tree view of files in the framework building system. These are not examples of a user interface for a business employee to interact with a business application software for handling tasks.

In summary, Lau teaches a first program that allows input of rules, which then generates a second program to carry out those rules. In contrast, Claims 1, 8 and 15 require that the computer program carrying out the rules to handle the tasks to be carried out by an employee, and also allow input of the rules by the user and then carrying out those rules.

With regard to the dependent claims 2-5, 7, 9-12, 14, 16-19 and 21, Lau fails to disclose these additional features as part of a computer program for handling tasks. These additional features include indicating which tasks are complete, receiving an event from an event queue, populating the queue with an event from the data component or the client component, and outputting tasks to a task assistant.

With regard to new claims 32-37, Lau also fails to disclose these new additional features as part of a computer program for handling tasks. These

Ser. No. 09/305,234
Amendment February 26, 2003


additional features include indicating the employee who is completing the task, and being responsive to an event of completing a task.

Conclusion

In view of the foregoing remarks, Applicants respectfully assert that the rejection is overcome and the claims are in condition for allowance. Should there be any matters of a formal nature to be clarified, please call the undersigned attorney in order to expedite allowance of this application.

Respectfully submitted,

Dated: 2-26-03



Marc V. Richards
Reg. No. 37,921
Attorney for Applicants

BRINKS HOFER GILSON & LIONE
P. O. Box 10395
Chicago, Illinois 60610
(312) 321-4200

APPENDIX

1. (Amended) A computer program embodied on a computer readable medium for [developing component based software capable of] handling tasks, comprising:

- a data component that stores, retrieves and manipulates data utilizing a plurality of functions; and

- a client component including:

- an adapter component that transmits and receives data to/from the data component,

- a business component that serves as a data cache and includes logic for manipulating the data, and

- a controller component adapted to handle events generated by a user utilizing the business component to cache data and the adapter component to ultimately persist data to a data repository,

wherein the client component is adapted for allowing a user to define tasks to be carried out by an employee that achieve a goal upon completion, allowing the user to input rules which dictate which of the tasks should be selected based on predetermined events, receiving at least one event, and outputting the task which is selected based on the received event in accordance with the rules.

8. (Amended) A computer program embodied on a computer readable medium for [creating a component based architecture capable of] handling tasks, comprising:

- a user interface form code segment adapted for collecting data from a user input;

- a business object code segment adapted for caching data;

- an adapter code segment adapted for transmitting data to a server;

and

- a controller component code segment adapted for handling events generated by the user interacting with the user interface code segment, providing validation within a logic unit of work, containing logic to interact with the business component, creating one or more business objects, interacting with the adapter component to add, retrieve, modify, or delete business objects, and providing dirty flag processing to notify a user of change processing;

wherein the computer program is adapted for allowing a user to define tasks to be carried out by an employee that achieve a goal upon completion, allowing the user to input rules which dictate which of the tasks should be selected

based on predetermined events, receiving at least one event, and outputting the task which is selected based on the received event in accordance with the rules.

15. (Amended) A computer program embodied on a computer readable medium for [creating a component based architecture for] allowing communication between a plurality of clients and a server in order to handle tasks, comprising:

one or more client components included with each client, each client component of each client adapted for communicating and manipulating data with a first data type,

wherein the client component is adapted for allowing a user to define tasks to be carried out by an employee that achieve a goal upon completion, allowing the user to input rules which dictate which of the tasks should be selected based on predetermined events, receiving at least one event, and outputting the task which is selected based on the received event in accordance with the rules;

one or more server components adapted for communicating and manipulating data with a second data type; and

one or more adapter components included with each client for translating data from the one or more client components to the second data type when communicating data from the client to the server and further translating data from the one or more server components to the first data type when communicating data from the server to the client.